Modeling and Evaluating Trust Network Inference *

Li Ding Pranam Kolari Shashidhara Ganjugunte Tim Finin Anupam Joshi

University of Maryland, Baltimore County
1000 Hilltop Circle, Baltimore MD 21250
(dingli1,kolari1,shashi1,finin,joshi)@cs.umbc.edu

Abstract

The growth in knowledge sharing enabled by the (Semantic) Web has made trust an increasingly critical issue. Based on explicit inter-agent trust relations, a trust network emerges on the (Semantic) Web in the knowledge sharing context. The concept of a trust network and its application to knowledge sharing have received recent attention but neither their structural properties (e.g. dynamics, complexity) nor inference mechanisms (e.g. trust discovery, trust evolution, trust propagation) have been well addressed. This paper formalizes trust network inference notions, providing both data and computational models, and suggests an evaluation model for benchmarking. The data model clarifies the data (context, restriction, output) used by trust network inference for knowledge sharing. It also elaborates trust network representation and articulates different types of trust. The computational model reviews graph theory and referral network interpretations of trust network inference and proposes a new one that treats trust network as an emergent property. This new model supports both trust evolution and trust propagation. The evaluation model describes metrics as well as methods to generate test scenarios and data. We argue that this approach is more customizable, flexible and scalable than traditional approaches such as public reputation systems and collaborative filtering.

1. Introduction

Our individual unique experiences lead to a diversity of knowledge. By sharing knowledge with one another, we greatly extend our understanding of the world. The Semantic Web will make large amounts of online information available in the form of simple statements to communities with both people as well as artificial agents. In such communities the following hold: (i) knowledge is sparsely distributed, but an individual can consult others to reduce his ignorance; (ii) there are many criteria for accepting a statement as true (e.g., social facts reflect the community consensus whereas subjective beliefs reflect an individual's judgments); and (iii) inconsistent knowledge may co-exist for a variety of reasons (e.g., individual disagreements, dishonest individuals, and erroneous belief judgments.)

Consider an abstract system of agents and objects, where each agent has beliefs over some objects and is ignorant of the rest. The problem is – how does an agent reduce its ignorance through knowledge sharing? A real world instance of this problem is "when shopping on an online store, a user may go to the rating service Bizrate (http://bizrate.com/) to check the store's reputation". In this case, the customer's ignorance is reduced by sharing rating information from Bizrate.

Traditional approaches to this problem include content filtering, social information filtering (collaborative filtering), reputation systems, Peer-to-Peer(P2P) systems, and referral systems [16]. Content filtering evaluates the relevance and usefulness of other agents' beliefs. Reputation systems build reputation for an object by summarizing agents' beliefs over it. Collaborative filtering assumes that similar agents have similar beliefs. The first three approaches are centralized systems, which might not scale well. P2P systems and referral systems, however, are designed for distributed environments and can be viewed as simple versions of a trust network. Trust is captured implicitly by the neighborhood relation in P2P systems. Referral systems even maintain and use trust for recommendation [15]. Unfortunately, all these approaches tend to underestimate trust, which is the key to knowledge sharing.

Trust can be used to reduce search complexity and to combine the beliefs of multiple agents. A *trust network* is essentially an online social network, where agents are interlinked by trust relations. Intuitively, trust is used to estimate the quality (e.g. the accuracy rate) of an agent's beliefs. Instead of having to know everything locally, agents share

* Partial research support was provided by DARPA contract F30602-00-0591 and by NSF by awards NSF-ITR-IIS-0326460 and NSF-ITR-IDM-0219649.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2005		**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Modeling and Evaluating Trust Network Inference	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Defense Advanced Research projects Agency,3701 North Fairfax Drive,Arlington,VA,22203-1714**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**12**	

knowledge based on their trust relations in a decentralized manner. In contrast to traditional approaches, a trust network has the following desirable features: *expressiveness* – the rich meaning of trust is explicitly captured; *flexibility* - agents can interpret trust by themselves; *scalability* trust supports unrestrained growth of the community; and *practicality*– trust can be locally evolved and propagated, and therefore facilitates the propagation of beliefs.

This work builds on our previous work on modeling trust [5]. Our goals are three fold: (i) to develop a rich framework for modeling and computing trust relationships in an open P2P system; (ii) to test, refine and evaluate the framework through simulations, and (iii) to "field test" the resulting framework as a component in one of more real world applications. This paper details the framework developed so far and our plans for evaluation. Candidate applications for field testing include several pervasive computing projects in our research group [11, 3, 12], an agent based trading simulator [21], and several new semantic web projects.

The rest of this paper is structured as follows: section two describes the *data model*, which formalizes the meaning of trust and explains data used in trust network inference; section three describes the *computational model*, which summarizes the primitive operations and existing interpretations of trust network inference; section four elaborates the operations and emergent properties in our emergence interpretation; section five proposes an *evaluation model*, which overviews our general framework, the evaluation metrics and the creation of test data (including real world data); section six briefs the experimental design; and section seven concludes our work.

2. Data Model

Through a trust network, agents form a peer-to-peer (P2P) system, sharing their knowledge and deriving beliefs collaboratively. We restrict agents as intelligent software agents and knowledge as belief status about objects. Figure 1 depicts the data model of trust network inference. The *environment* (i.e.,a P2P system) is characterized by context and restriction, which are partially known by an agent. Trust network inference takes input from both environmental knowledge and agents' trust knowledge, updates its trust knowledge and outputs the estimated beliefs. The *expected belief* is what the agents really want, and is approximated by the estimated belief output by trust network inference. In the rest of this section, we elaborate the boxed data: "context", "restriction", "estimated belief", "expected belief" and "trust" respectively.

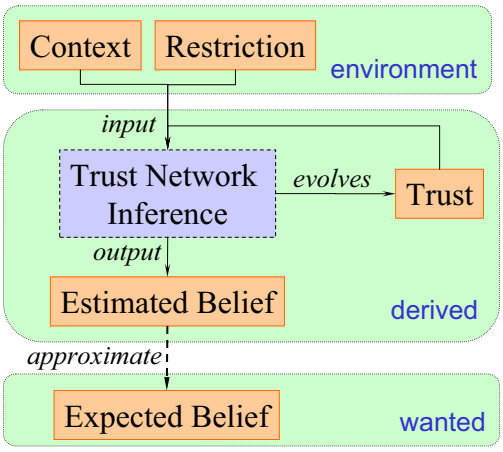

Figure 1. Trust Network Inference Data Model

2.1. Context

The *context* is part of the input of trust network inference. It describes the general configurations of a P2P system of agents, e.g., "how many agents and objects are included", and "how knowledge is distributed". The P2P system is defined by having N distinctive agents $A = \{a_1, a_2, ...a_N\}$ and M distinctive objects $O = \{o_1, o_2, ...o_M\}$. The range of belief status varies in different applications: Boolean is used for logical inference purpose, and real is used for rating purpose. We assume that the objects are propositions, and that the range of belief status is $SVB = \{unknown, true, false\}$, so that an agent may have three belief status about a proposition. The knowledge distribution of the P2P system is represented by a *belief matrix*

$$belief : A \times O \rightarrow SVB \quad (1)$$

where $belief_{io}$ refers to an agent a_i's *personal belief* about an object o_o. An agent a_i's knowledge base is a vector $KB^{[i]} = \{belief_{io} | 1 \leq o \leq M\}$.

Relevant objects can be grouped by a set of *domains* $D = \{d_1, d_2, ...d_L\}$. Intuitively, belief status about relevant objects is collected with similar accuracy and used for similar purpose. Domain information provides a short digest about an agent's belief distribution, so the value of L should be small to avoid complexity issues. In addition, it is not practical to assume that all objects share a single domain. For example, consider two propositions about a website: (o_1) "website X has good page design" and (o_2) "website X provides secure payment mechanism". Since people often use them for different purpose, it is better to assign them to two different domains, i.e., "information quality" for o_1 and "security" for o_2.

2.2. Restrictions

Knowledge sharing is motivated and restricted by bounded resource, such as process time, communication bandwidth, and memory size. In trust network inference, we focus on the restrictions that directly affect inter-agent communication, namely "knows matrix" and "cost matrix".

The directed "knows" relation restricts the communication channel between agents. It restricts an agent's neighborhood (the agents it can directly send a query). Intuitively, the "knows" relation functions like an address book, i.e., for any two agents to interact, one agent should know the other agent's address before it initializes the communication. A *knows matrix* collects the "knows" relation, which is represented by:

$$knows : A \times A \rightarrow \{false, true\} \tag{2}$$

where $knows_{ij}$ means agent a_i knows agent a_j's communication address. The "knows" relation is not transitive: given $knows_{ij} = true$, $knows_{jk} = true$ and $knows_{ik} = false$, a_i knows a_k unless a_j introduce a_k to a_i.

The directed "cost" relation imposes cost restriction to agent communication. It enables agents to do cost/benefit analysis before they communicate. Intuitively, agents have to pay for communications, e.g., money spent in long distance phone call, and time spent in waiting for email reply. There is also a social cost to communication, since it puts a burden on the query agent, and the level of burden one can put on a given agent depends on many social factors. Even with the web infrastructure, communication cost is nontrivial: sharing beliefs with a colleague at work is much cheaper than consulting with a busy expert in another country. The *cost* relation can be represented by:

$$cost : A \times A \rightarrow R+ \tag{3}$$

where $cost_{ij}$ means the average cost when agent a_i communicates with agent a_j. It is notable that the cost matrix is not symmetric, i.e., $cost_{ij}$ may not equal to $cost_{ji}$.

2.3. Estimated Belief and Expected Belief

The *estimated belief* is derived from trust network inference, and the *expected belief* is the ideal target for trust network inference. We assume that the expected beliefs can be derived given the complete global knowledge. However, agents in P2P systems can only obtain knowledge from limited peers through trust network inference. By evolving trust network, agents may derive the estimated belief closer to the expected belief.

The expected belief ϕ is defined similar to belief, where ϕ_{io} means agent a_i's expected belief over object o_o. It can be derived from a weighted aggregation of all agents' beliefs:

$$\phi : A \times O \rightarrow SVB \tag{4}$$

$$\phi_{io} = \diamondsuit_j [w_{ijo}, belief_{jo}] \tag{5}$$

where w_{ijo} is weight assigned by a_i over a_j's belief on o_o. $\diamondsuit$ denotes the aggregation function that combines the weighted beliefs.

2.3.1. Interpretations of expected belief. It is notable that agents may have different interpretations and weighting schemas for the expected belief. Commonly used interpretations are listed as the following:

Local isolated. This interpretation assumes that an agent fully trusts itself while never trusts the other agents. An agent a_i's expected belief ϕ_{io}^{local} about object o_o is the same as $belief_{io}$.

Global uniformly weighted. This interpretation assumes complete global knowledge and treats all agents equally. The weighting schema is shown in equation 6.

$$w_{ijo}^{g-uniform} = 1 \tag{6}$$

Global weighted. This interpretation assumes enough global knowledge and treats all agents differently by their global reputation. Global knowledge is "enough" when it includes the knowledge from all agents' with non-zero weight. The global reputation is denoted by R_j, which comes from certain reputation system. . The weighting schema is shown in equation 7.

$$w_{ijo}^{g-weighted} = R_j \tag{7}$$

Collaborative. This interpretation assumes that agents decide weighting schema themselves. Given an object o_o, each agent a_i weights another agent a_j's beliefs based on a_i's evaluation about a_j's quality E_{ijo}. By assuming that similar agents have similar beliefs, collaborative filtering derives E_{ijo} from the pair-wise similarity between agents. The weighting schema is shown in equation 8.

$$w_{ijo}^{collaborative} = E_{ijo} \tag{8}$$

2.3.2. Aggregation functions. Aggregation functions combine agents' beliefs into one belief. The range of belief value, i.e., SVB, determines the selection of aggregation function. "Majority consensus" functions are well-known for handling beliefs with discrete value, and numerical functions are useful for handling beliefs with continuous value (e.g. integer, real number).

We first summarize the numerical approaches for the continuous belief aggregation. Let $SVB = [L, H]$, where L and H are the lower bound and higher bound respectively,

and $SVB \subset R$. According to [6, 14], there are four interpretations: average, median, max and min.

We then formalize two majority consensus approaches for discrete belief aggregation, namely "simple majority" and "entropy based majority". Let $SVB = \{v_0, v_1, v_2, ...v_Z\}$, where v_0 represents "unknown" and the rest v_i are independent and non-comparable belief values. The summed weight W_{io} is defined by equation 9, where S be subset of SVB.

$$W_{io}(S) = \sum_{a_j \in A,\ belief_{jo} \in S} w_{ijo} \tag{9}$$

The *simple majority* aggregation function (equation 10) uses an empirical value α_k (our previous experiments use 0.5) to control the consensus result based on context bias.

$$\diamondsuit_{io}^{majority} = \begin{cases} v_k & if\ \frac{W_{io}(\{v_k\})}{W_{io}(SVB)} \geq \alpha_k \\ v_0 & otherwise \end{cases} \tag{10}$$

The *entropy based majority* aggregation function (equation 13) also uses an empirical value α to control error. An entropy [10] reflects the degree of disorder of the beliefs collected from all agents. Only when $entropy$ is low enough, consensus might be reached.

$$entropy_{io}(S) = \sum_{v_k \in S} -\frac{W_{io}(\{v_k\})}{W_{io}(S)} \log_2 \frac{W_{io}(\{v_k\})}{W_{io}(S)} \tag{11}$$

$$\begin{aligned} gain_{io}(S, v_k) = {} & entropy_{io}(S) \\ & -\frac{W_{io}(S-\{v_k\})}{W_{io}(S)} entropy_{io}(S - \{v_k\}) \end{aligned} \tag{12}$$

$$\diamondsuit_{io}^{entropy} = \begin{cases} v_0 & if\ entropy_{io}(SVB) \leq \alpha \\ \arg\max_{v_i \in SVB} (gain_{io}(SVB, v_i)) & otherwise \end{cases} \tag{13}$$

2.4. Trust Data

In a P2P system, *trust* characterizes the directed pairwise inter-agent trust relation. It can be viewed as a special type of belief, and can be interpreted as "the estimation of an agent's belief quality", or "the similarity between two agents". The global trust state of the entire agent society can be captured by a *trust matrix*, which is represented by:

$$trust : A \times A \times D \rightarrow SVT \tag{14}$$

where $trust_{ijd}$ means how much agent a_i trust agent a_j's belief on objects in domain d_d. SVT refers to the range of trust values (see section 2.4.1). Normally, an agent a_i only maintains its own trust knowledge, $\{trust_{ijd} | 1 \leq j \leq N, 1 \leq d \leq L\}$, and obtain the other agents' trust knowledge through agent communication. In addition, trust is defined on domains but not objects due to resource bounds. By assuming all objects belong to a single domain, the trust matrix can be simplified to a 2D matrix. However, this simplification may not work well in practice, since not all objects are relevant enough to belong to one domain.

2.4.1. Trust value. The value of trust reflects the confidence over the trust knowledge. The range of trust, SVT, can be either Boolean or numeric. Boolean trust is always the assertive decision derived from either logical inference or machine learning results. Numeric trust value can capture the uncertainty of trust more accurately; therefore, it is suitable for trust network inference. We note that no agreements have been made [7] on the exact numerical trust representation, especially when distrust is considered. So we identify four existing type of numeric trust value schema as the following:

1. Golbeck et al. [6] suggest a graded trust which has nine grades ranging from absolutely distrust to absolutely trust. This idea is a simple extension of graded ratings in reputation system, and human users can easily understand and annotate trust ratings.

2. Richardson et al.[14] and Guha et al. [13] suggest using real value trust within $[0, 1]$, where 0 means fully ignorance and 1 means fully trustworthy . Distrust is maintained by maintaining trust over the negation of beliefs.

3. Josang [1] proposes subjective logic based trust and Yu and Singh [18] prefer a Dempster-Shafer Theory based trust to represent trust as a triplet *(trust, distrust, ignorance)*, where ignorance is used to capture the uncertainty of trust due to lacking of experiences. However, there could be another component called "untouched", which shows the proportion of example space (all observations) to the entire domain space. For example, ratings based on 100 reviews are less trustworthy than those based on over 10,000 reviews in Bizrate.com; "Familiar with football" does not necessarily imply "familiar with sports"; and misunderstanding often come from unbalanced observations. The "untouched" component can be removed only when either there is no unsolicited items in the domain, or assume the solicited experience has the same distribution as the unsolicited experience.

4. Our previous work [5] used a real value trust within $[0, 1]$, where 0 means fully distrust, 0.5 means fully ignorant, and 1 means fully trust. We adopt this representation for two reasons: (i) trust knowledge evolves as a Markov process, i.e., agents adjust trust value only based on current trust value and newest observations; (ii) the three trust state (trust, ignorance, distrust) are exclusive, and ignorance is unavoidable when moving between trust and distrust.

2.4.2. Types of trust. Simply knowing "a_i trusts a_j in domain X" does not fully capture the meaning of trust, and we should also consider the provenance and usage of trust knowledge. Therefore, we classify two categories of trust in knowledge sharing context: (i) *referral trust* reflects an agent's estimation about the quality of the other agents' knowledge. It is derived by an agent itself and used to direct query communication. Since it evolves dynamically, we adopt the fourth trust value schema defined in section 2.4.1 (ii) *associative trust* reflects the similarity between two agents. It is derived by comparing two agents, and its reliability depends on the number of observations and the distribution of evidence. Since it does not introduce distrust, we adopt the second trust value schema defined in section 2.4.1.

It is notable that trust itself can be used to propagate trust. Equation 15 depicts how agent a_i derive trust to a_j via its trusted agent a_k.

$$trust_{ij}^{X} = trust_{ik}^{Y} \circ trust_{kj}^{Z} \tag{15}$$

where $\circ$ is a concatenation operator, which is commonly a multiply operator.

Based on [15, 5, 13], we identify five types of trust and their trust propagation mechanisms as the following:

Domain Expert Trust (DET) is referral trust that evaluates the quality of an agent's domain knowledge. $trust_{ijd}^{DET}$ refers to an agent a_i's estimation about the quality of agent a_j's belief over any objects in domain d_d. Intuitively, DET is not transitive, but DET may imply RET (see next item) on the same domain. Figure 2 depicts agent 'A' have DET of 0.8 to agent 'U', so 'A' might infer its RET of 0.7 to 'U'.

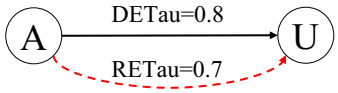

Figure 2. Propagate trust using DET

Recommendation Expert Trust (RET) is referral trust that evaluates an agent's trust knowledge. $trust_{ijd}^{RET}$ refers to an agent a_i's estimation over the quality of agent a_j's trust knowledge in domain d_d. In real world, the domain used in RET is often much wider than DET, e.g. CNN is a domain expert only in news area, while Google.com is a recommendation expert in almost any area. Moreover, RET in transitive according to its definition, so it can be used to propagate both DET and RET. Figure 3 depicts an agent 'U' having a DET of 0.9 to another agent 'V'. Agent 'A' who has an RET value of 0.8 can infer a DET value of 0.72 to agent 'V'.

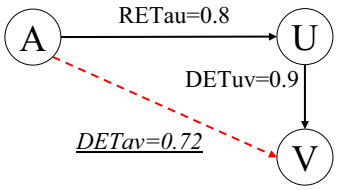

Figure 3. Propagate trust using RET

Similar belief trust (SBT) is an associative trust that evaluates the similarity of two agents' domain knowledge. $trust_{ijd}^{SBT}$ refers to the similarity from agent a_i's beliefs to agent a_j's beliefs within domain d_d. Intuitively, SBT clusters information providers, and it can be used to propagate DET. Figure 4 depicts two agents 'U' and 'V' who have similar beliefs $\{b_1, b_2, ..., b_k\}$, so they have a high SBT value of 0.9. This enables an agent 'A' who has a DET trust value of 0.8 to infer a DET value of 0.72 to 'V'.

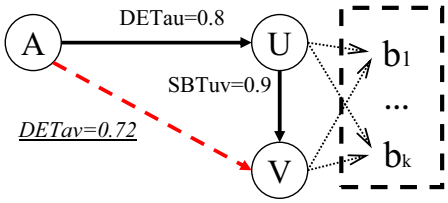

Figure 4. Propagate trust using SBT

Similar trusting trust (STT) is an associative trust that evaluates the similarity of two agents' trust knowledge. t_{ijd}^{STT} refers to the similarity of agent a_i and agent a_j's referral trust to the other agents within domain d_d. Intuitively, STT clusters trustors (agents who maintain trust knowledge), and it can be used to propagate both DET and RET. Computing STT needs trust knowledge from only two agents. Figure 5 shows two agents 'U' and 'V' with similar trust towards agents $\{a_1, a_2, ..., a_n\}$. If 'U' has a DET trust value of 0.8 towards a new agent 'A', then 'V' can infer a trust value of 0.72 to 'A'.

Similar cited trust (SCT) is an associative trust that evaluates the similarity of how two agents are trusted. $trust_{ijd}^{SCT}$ refers to the similarity of agent a_i and agent

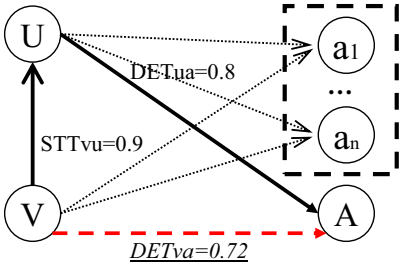

Figure 5. Propagate trust using STT

a_j are trusted by other agents within domain d_d. Intuitively, STT clusters trustees (agents who are trusted) by their reputation, and it can be used to propagate both DET and RET. Reliable SCT requires trust knowledge from a large population of agents. Figure 6 shows a set of agents $\{a_1, a_2, ..., a_n\}$ who have similar trust towards agents 'U' and 'V'. So an agent 'A' who has an RET trust value of 0.8 to 'U' can infer an RET trust value of 0.72 to agent 'V'.

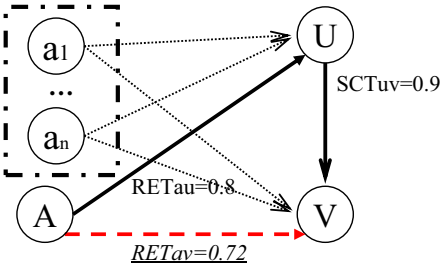

Figure 6. Propagate trust using SCT

3. Computation Model

The operations involved in trust network inference are not limited to propagating and combining beliefs through trust. We also need to consider how trust knowledge is discovered and evolved. In the rest of this section, we propose a general computation model which consists of five primitive trust operations. We then compare how three interpretations fit in this model.

3.1. Primitive Trust Operations

Since trust is the hypothesis learned from past experiences, trust network inference should consider both creating and using a trust network. We identify five primitive operations as the following:

1. **Trust discovery.** Though we can evolve trust network without any prior trust knowledge, it is highly desired to discover trust knowledge from publicly accessible information on the (Semantic) Web. Prior trust knowledge can be derived from many sources: (i) (Semantic) Web documents that contains inter-personal relations, e.g. FOAF personal profile, DBLP co-authorship and Epinions.com "web of trust"; (ii) link structure of the (Semantic) Web, e.g. how one's homepage links to another person; and (iii) correlations, especially similarity analysis results, e.g. co-belief (having quite a lot overlap in beliefs) and co-cited relations between individuals.
2. **Trust evolution.** Evolving trust enables an agent to derive personal trust knowledge from its own experience. Unlike the discovered trust knowledge, evolved trust knowledge is more reliable because it is based on first hand experiences. The intuition of trust evolution is: agents assume consensus results as correct belief to label their query experiences with other agents, and then use these labeled experience to evolve their trust knowledge. In practice, DET is learned first, and then RET is learned based on DET.
3. **Trust propagation.** Through an existing trust network, we need to propagate trust effectively and correctly for many purposes: to derive the missing edge in trust network, to find relevant information sources efficiently, and to aggregate beliefs from multiple agents. Based on the discussion in section 2.4.2, both referral trust and associative trust can be used to propagate referral trust. An important issue is how to concatenate and aggregate trust knowledge during trust propagation.
4. **Trust directed query.** Trust can help navigating the trust network effectively. Given the context, restriction and trust knowledge, agents need to direct queries collaboratively to a limited amount of the most helpful agents. In addition, upon receiving a query, an agent may reply its belief or refer other agents based on its referral policy.
5. **Trust based belief aggregation.** Having collected the beliefs from peer agents, an agent needs to aggregate these beliefs to derive the estimated belief. Two issues should be considered: (i) choosing appropriate aggregation function, and (ii) converting trust to Boolean value (it is also called "rounding" by Guha et al. [13] and emphasized as a non-trivial problem).

3.2. Interpretations of Trust Network Inference

Table 1 briefly compares three interpretations of trust network inference (the first two are from existing work

	Graph theory based	Referral network	Emergence
Context data	belief matrix	expertise vector	belief matrix
Restriction data	no	neighbor list and acquaintance list	knows matrix
Trust data	referral trust, SCT, STT	DET, RET	DET, RET, SBT, STT, SCT
Expected belief	not required	use interest vector	not required
Trust discovery	no	no	inferred from FOAF data
Trust evolution	no	yes, feedback	yes, Marcov process
Trust propagation	yes	no	yes
Trust based query	follows all possible paths	follows only relevant paths	follows only trusted paths
Trust based belief aggregation	max, min, average	user determined	simple majority consensus

Table 1. A comparison of three trust network inference interpretations

and the last one is proposed by us) under the computation model. The remainder of this section and section 4 will discuss them in detail.

3.2.1. Graph theory based interpretation. Graph theory has been used to interpret trust network inference [6, 14, 13]. The interpretation assumes a single-domain transitive directed trust network. The data model uses: a belief matrix B, a 2D trust matrix T for beliefs, another 2D trust matrix D for the negated beliefs, and a trust propagation matrix R. It does not differentiate DET and RET. The computation model involves: (i) trust propagation, R is created by the following functions [13]

$$\begin{aligned} M &= f(T, D) \\ R &= \alpha_1 M + \alpha_2 M^T M + \alpha_3 M^T + \alpha_4 M M^T \end{aligned} \tag{16}$$

where α_i weight the contributions from different type of trust, and f(x,y) determines how to propagate distrust. R^k refers to the propagated trust at k^th iteration; (ii) Trust based belief aggregation, the estimated belief $E^{(k)}$ is derived by equation 17.

$$E^{(k)} = \begin{cases} B & if\ (k = 0) \\ E^{(k-1)} \cdot R & if\ (k > 0) \end{cases} \tag{17}$$

The issues with this interpretation are:

- This model does not support "strong trust path" intuition, i.e., a trust path is valid unless all steps on it are highly trusted. In figure 7, graph theory based interpretation treat both paths P1 ($a \rightarrow b \rightarrow d$) and P2 ($a \rightarrow c \rightarrow d$) the same, and the propagated value is 0.38 ("max" is used as path aggregation function). However, real world users may disregard P1 since ($a \rightarrow b$) is weak for propagating trust, and will only consider P2 as a valid trust path.
- The propagation matrix R consists of both referral trust and associative trust. However, effective associative trust requires global knowledge and stable trust knowledge. In addition, according to the discussion in section 2.4.2, STT should be put to the left of $E^{(k-1)}$ in equation 17.

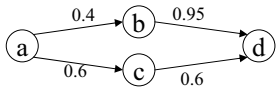

Figure 7. Trust propagation

3.2.2. Referral network interpretation. Social theory also has been used to interpret trust network inference [8, 15, 20]. This interpretation assumes a multi-domain directed referral social network. The data model uses: an agent's domain knowledge is stored in expertise vector E, the expected belief is stored in interest vector I, the communication restriction is captured by its neighbor list and acquaint list, its DET is maintained as "estimated expertise", and its RET is maintained as "sociability". The computation model involves: (i) trust evolution, where an agent evolves its trust over the solicited agents based on its judgment about their answer; (ii) trust based query, where an agent starts from its neighbors and searches through referral network for "good" answer collaboratively (a participant agent may reply with a confident answer as well as refer another expert).

The issues with this interpretation are:

- The answer evaluation mechanism is only based on the relevancy of an answer, which might be vulnerable to malicious agents providing relevant but incorrect answer.
- It may not be appropriate to assume that an agent already knows the expected belief.
- Reply with answer and refer other experts may not be exclusive.

3.2.3. Emergence interpretation. Emergence interpretation roots from referral network interpretation. It also makes

the following improvement: it allows agents to be more personalized; it does not assume that agents know expected belief and allows agents to derive estimated belief by consensus, it allows the existence of inconsistent knowledge, and it supports automatic trust evolution. The data model includes: each agent knows its domain knowledge, resource restrictions, and its policies for local interaction. Prior trust knowledge can be added as reference in the absence of self-derived trust knowledge. The computation model includes all five primitive trust operations. We leave the details of this interpretation to the next section.

4. Emergence Interpretation

The emergence interpretation concerns both the maintenance and usage of trust network inference. In P2P system, trust network is maintained by individual agents at micro level, and it is used (i.e., queried) as one entity at macro level. Macro level properties associating with a trust network could be the graph structure, overall query accuracy, agent interaction pattern. In the rest of this section, we first discuss micro level activities and then discuss macro level properties.

4.1. Local Agent Interaction

In knowledge sharing P2P system, local agent interactions are primarily query activities. A *query activity* is defined as a tuple (i, k, est) , which means that agent a_i queries the belief about object o_k and then concludes the estimated belief est. The *estimated belief* derived by consensus can be used as correct answer in absence of the expected belief.

Within the query activity, a_i may consult zero or more other agents' for their beliefs over o_k, and then determine their contributions by comparing their beliefs with est. A *contribution* shows importance of a consulted agent's belief to the estimated belief. It is defined as a tuple $(i, k, j, type, weight, belief)$ where a_j contributes a $belief$ in a certain $type$ with $weight$ to a_i's query on o_k. The "type" of a contribution indicates what part of a_j's knowledge is used, such as belief over o_k, and referral trust about another agent. The "weight" of a contribution indicates both the estimated strength and the correctness of a_j's knowledge, and it is defined within $[-1, 1]$, where the strength is determined by the absolute value and the correctness is determined by the sign, e.g. 0.8 corresponds to a correct belief with strong contribution and -0.1 corresponds to an incorrect belief with weak contribution. The "belief" is a_j's real belief.

4.2. Trust Evolution

Trust evolution focuses on evolving referral trust through agent query interactions. Intuitively, trust evolution brings frequently used experts closer, and it also brings agents with similar interest closer. In our practice, DET and RET are evolved differently through the following steps:

- *Derive the correct belief.* Correct belief is used to determine the correctness of the beliefs contributed by other agents. By running majority consensus in trust based belief aggregation, we use the estimated belief as the correct belief when not knowing the expected belief.
- *Evaluate the quality of contributed belief.* Trust is evolved differently based on "boostFactor", which shows the quality of contributed belief. "boostFactor" ranges within [-1,1]. Its sign indicates whether an increase or decrease of trust, and its absolute value indicates how much should trust value change. Table 2 shows our practice in trust using Boolean value.

contributed belief / correct belief	true	false	unknown
true	1	-1	0
false	-1	1	0
unknown	0.2	0.2	0

Table 2. BoostFactor example

- *Evolve DET.* We adopt stateless Markov learning model, which use only current state to derive the next state. Given all queries are about the same domain, the trust evolution runs according to a recursive function as below:

$$t_{(n)} = f(t_{(n-1)},\ boostFactor,\ weight) \qquad (18)$$

where $t(n)$ refers to the trust value after an agent's n^{th} query activity. "boostFactor" is determined in table 2, and "weight" is the weight of the participant agent's contribution. We further suggest two implementations of f:

$$f(x, bf, w) = x^{(1-bf \cdot w)} \qquad (19)$$

$$f(x, bf, w) = \begin{cases} x(1 + bf \cdot w) & if\ bf < 0 \\ x + (1 - x)bf \cdot w & else \end{cases} \qquad (20)$$

- *Evaluate referrers' contributions.* A referrer's contribution is determined by the contributions of the domain experts it recommended.

- *Evolve RET.* This is a complex issue where many heuristics can be used [19]. By only evolving RET to the referrers who directly recommended a domain experts, an agent can reduce the length of referral path.

4.3. Trust Propagation

According to the discussion in section 2.4.2, an agent can build a local trust propagation graph which is similar to referral graph [19]. Unlike the evolved referral trust, propagated trust is temporarily derived for a certain query activity, and it will not modify an agent's personal trust knowledge directly. In addition, we need to have sufficient confidence over the propagated trust. We adopt the following heuristics to propagate trust effectively and correctly:

- *"Strong trust path"*. Each edge in the local trust propagation graph should be highly trusted; therefore, no "weak link" will be part of a trust propagation path.
- *"Maximum trust path length"*. Since complete trust is a rare phenomenon, the longer a trust propagation path is, the more risk we will have. We can either use a threshold to skip less trusted propagated trust or simply use a threshold to skip longer paths.
- *"Social distance ordering"*. In order to avoid referral cycle, an agent propagates trust in breath first search style. It does not propagate trust from agents with longer social distance to agents with shorter social distance.

4.4. Trust based Query

Trust based query in emergence interpretation is similar to that in referral network. There are two important issues: "how an agent selects a set of agents for consultation" and "how an agent answers a consultation".

In a query activity, an agent needs to control search complexity and avoid cycles. Instead of delegating query to other agents through query flooding, referral network approach lets the consulting agent initiate all communications. Singh et al. suggested an approach with knowing expected belief [15, 20], and our previous work [5] suggested a majority consensus approach without knowing expected belief. A consulting policy should consider how to select the agents to be consulted and how to query them in an appropriate order [17].

Upon receiving a consultation, an agent responds based on its reply policy. The reply can be the agent's real belief, or its recommendation to other agents, or both. Normally only strongly believed beliefs can be returned. The recommended agents can be further filtered based on their relevancy to the query and their RET value [20, 5, 17].

4.5. Trust based Belief Aggregation

Belief aggregation depends on the consulted beliefs retrieved by trust based query and the propagated DET derived by trust propagation. We can use the aggregation functions in section 2.3.2 to derive the estimated belief.

4.6. Emergence Properties

The observed emergence properties are mainly trust network graph structure properties:

- *Degree distribution.* Guha et al. [13] reported that the indegree and outdegree distribution of trust network in Epinoins.com follows power law, but its exponent parameter is not common in comparison with the other graph structures on the Web. The observation shows that experts are not very popular. Our hypothesis for this difference is: even experts may have quite a lot of ignorance while people may use different trust evaluation criteria.
- *Bow tie structure.* Guha et al. [13] also reported that trust network contains strongly connected component (SCC) and forms a bow tie structure [2]. Our hypothesis for this property is: experienced users not only have a lot of domain knowledge but know the other helpful experts; therefore, they are better trusted since they always offer good news; in addition, new users may start by trusting the experienced users.
- *Emergence of DET experts and RET experts. DET experts* in trust network is essentially agents with high indegree (an authority). They emerge when enough agents in the network has positive experiences with them. *RET experts* in trust network are essentially agents with high outdegree (a hub). They emerge when many agents need referral services, i.e., they can't afford the cost of memorizing all useful domain experts.
- *Convergence of trust propagation graph.* We propose a hypothesis that: for each agent, given enough experience and assuming static agent knowledge, their trust propagation graph will converge to a stable state.

5. Evaluation Model

In order to compare and benchmark the different interpretations of trust network inference, we propose a evaluation model, which is depicted in figure 8. In general, evaluation runs in the three consecutive steps: *configuration*, which initializes the agent society; *inference*, which allows the analyzer to simulate agent query activities; and *evaluation*, which reports the statistics of the simulation results.

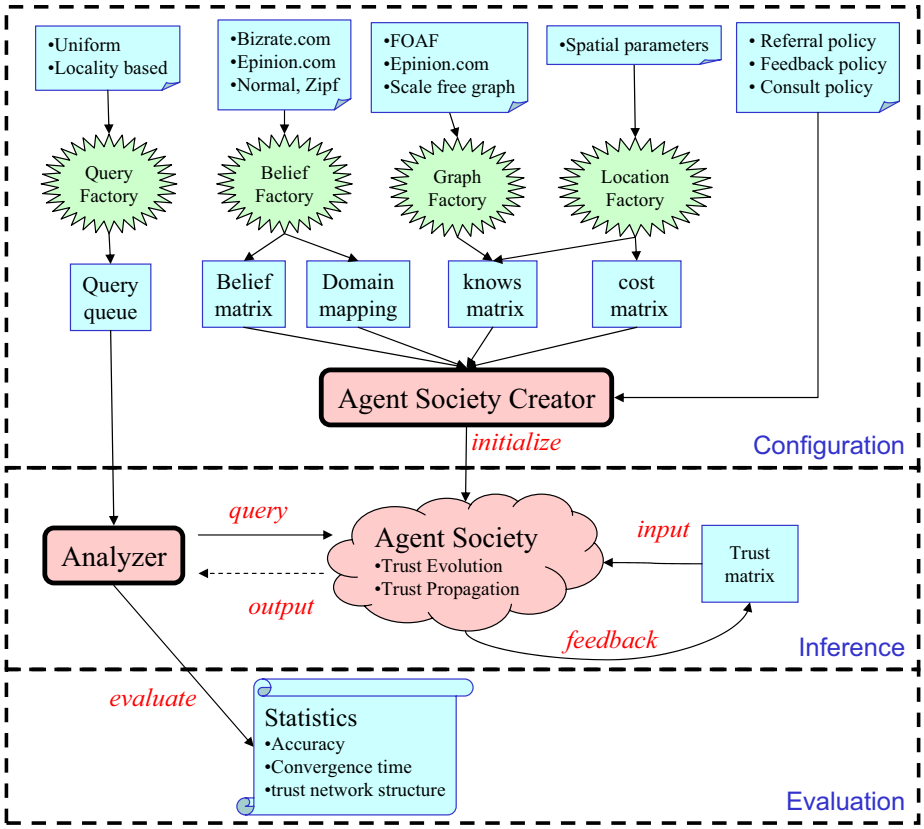

Figure 8. Trust network inference evaluation model

5.1. Evaluation Metrics

We use evaluation metrics to characterize the distribution of data and the performance of operations in trust network inference.

5.1.1. Belief quality. The quality of a real belief $belief_{io}$ is derived by comparing it with the corresponding expected belief ϕ_{io}. For Boolean trust, table 3 shows three types of exclusive belief qualities: "correct", "error", and "ignorance".

$belief_{io}$ / ϕ_{io}	true	false	unknown
true	correct	error	error
false	error	correct	error
unknown	ignorance	ignorance	correct

Table 3. Belief quality

5.1.2. Belief vector quality. Given a vector of beliefs, e.g. an agent's knowledge base, we use three features to characterize its quality: (i) $P(I)$ – the proportion of beliefs with "ignorance" quality, (ii) $P(C)$ – the proportion of beliefs with "correct" quality, and (3) $P(E)$– the proportion of beliefs with "error" quality. The two extremes of $P(I)$ are "knowledgeable" and "ignorant"; those of $P(C)$ are "wise" and "unwise"; and those of $P(E)$ are "deceptive" and "honest". These three features are related by equation 21.

$$P(I) + P(C) + P(E) = 1 \tag{21}$$

5.1.3. Belief matrix distribution. Belief distribution may directly affect the result of majority consensus, and therefore affect the trust evolution result. The distribution of a belief matrix B can be evaluated by the following benchmarks: (i) matrix level benchmarks, such as sparsity, which shows the proportion of ignorance in B (see equation 22); and (ii) vector level benchmarks, such as the distribution of $P(I)$, $P(C)$, $P(C|E)$. We can evaluate both row vector and column vector of B: the quality of a row vector in B is called *user quality*, which refers to the belief distribution of an agent's knowledge base; the quality of a column vector in B is called *review quality*, which refers to the dis-

tribution of beliefs about a certain object.

$$\text{sparsity}(B) = \frac{number\ of\ \text{“}ignorance\text{”}\ beliefs}{number\ of\ all\ beliefs} \quad (22)$$

5.1.4. Query distribution. A query activity selects an agent to derive beliefs about an object. A query queue is composed of a sequence of pairs (agent, object). We can derive query frequency distribution for agents and object. In addition, we can analyze the co-occurrence distribution, so as to determine the dependency between agents and objects.

5.1.5. Query result distribution. Given the query result, we can evaluate the performance of trust network inference, e.g. “the quality of estimated beliefs”, and “how many agents are consulted per query”. Richardson et al. [14] suggested *precision* and *recall* for static trust network. Their benchmark shows how trust network helps agents to reach correct belief in comparison with the case of agents having global knowledge.

5.1.6. The convergence of trust network. Richardson et al. [14] suggests that a trust network converges when the trust knowledge of all agents stops changing. However, this criterion might not fit for dynamic systems. Based on emergence interpretation, we hypothesis that a trust network converges when the distribution of query result converges. Specifically, when the average accuracy of query result converges, we believe the trust network will converge.

5.1.7. Trust network graph structure. Guha et al. [13] pointed out two important graph structure properties: *degree distribution* and *strongly connected component*. We expect to observe Zipf’s/Power distribution in both in-degree and out-degree, and Bow-tie structure in the evolved trust network.

5.2. System Configuration

Our evaluation model intends to provide both real world data and synthesized data to benchmark trust network inference interpretations.

5.2.1. Belief matrix. It is difficult to obtain an individual’s beliefs in real world because of privacy issues. However, many reputation systems collect the overall consumer ratings over objects like products, services, and websites. They may also supply some anonymous personal ratings (e.g. Bizrate.com, Amazon.com, and IMDB.com), or some named ratings with textual reviews (e.g. Epinions.com). Therefore, real world belief data exists in two forms: (1) the average and the distribution of beliefs over an object; (2) the complete belief matrix.

We synthesize a belief matrix in two steps. (1) Generate vector quality. Vector quality can be directly obtain from real world data, or synthesized from distribution. We determine the three features in a certain order: first, let $P(X)$ be the primary feature and follow a certain distribution, such as normal distribution. Second, let $P(Y|X)$ follow a different distribution and compute $P(Y) = P(Y|X) * P(X)$. Finally compute $P(Z)$ using equation 21. In this approach, any one of $P(I)$, $P(C)$, $P(E)$ can be the primary feature, and both Zipf/Pareto distribution and normal distribution can be used. (2) Initialize belief matrix with vector quality. We can randomly assign values for each cell in belief matrix without violating the distribution given by the vector quality.

5.2.2. Knows matrix. Real world “knows” knowledge can be obtained from many places, such as address book of e-mail clients, social network websites (e.g. orkut.com, Epinions.com), online FOAF files (we have crawled millions of them), or web scraping results (e.g. co-author relation from DBLP).

The synthesized “knows” matrix could be a random network, a scale-free network, or a small world network [4]. Moreover, spatial distance can be used to initialize a “knows” matrix, where agents know one another only when they are geographically close enough.

5.2.3. Trust matrix. Currently, the real world trust knowledge is collected by websites, such as the “web of trust” at Epinions.com and the trust network at advogato.org. Privacy issues make most of these data not publicly available.

The synthesized trust matrix comes from two sources: (i) the discovered social networks may imply certain trust relation between individuals, so we may combine these social networks and use the implied trust to create a trust network; and (ii) we can also evolve trust network in emergent way in P2P system, where no prior trust knowledge is required.

5.3. Query Queue Generation

After agent society has been configured, the analyzer can simulate queries using pre-generated query queue, which is a sequence of query activities. A simple approach is to uniformly select the agent and object and then form a query. We can also select agents and object using certain distributions. Based on Kittock’s argument of non-uniform interaction probability [9], we can select agents by their interaction distribution. Also, for each agent, we can select objects by the agent’s interest vector.

6. Experiments

Our ongoing experiments consist of three parts. (i) A general purpose trust network evaluation package called “TrustWeb”. This will provide both native java interface for

local simulation, and web service based interface for distributed simulation and application. (ii) Evaluation results for trust evolution and discovery. We will show how trust network emerges through evolution, and the role of context, restriction, and query distribution in this process. We will also show how trust network can be discovered from real world data. (iii) Evaluation results for trust propagation, trust based query and belief aggregation. We will show how knowledge distribution, query distribution and the structure of trust network affect the query result distribution.

7. Conclusion

Trust networks are critical for knowledge sharing in open, dynamic and large-scale agent societies on the (Semantic) Web. Trust can be used to guide queries effectively to the most helpful agents and adopt knowledge selectively from multiple sources. These advantages have led to various interpretations of trust network inference.

In this paper, we study the data and computational models involved in trust network inference and propose an evaluation model to compare the effectiveness of existing interpretations. We propose a generic *data model* that enumerates the inputs (context, restriction), the output (the estimated belief) and the target (the expected belief) of trust network inference. The data model details why trust is domain specific in the real world and identifies several types of trust in knowledge sharing context. We also summarize primitive operations in a generic *computational model*, and compare existing interpretations of trust network inference, based on graph theory, referral network and emergence. Our emergence interpretation enables agents to both discover and evolve trust knowledge for trust based operations. We are testing our evaluation model over these interpretations using both real world and synthetic data. This will in turn facilitate the adoption of trust network inference by real world applications in various domains.

References

[1] A.Josang. An algebra for assessing trust in certification chains. In *the Network and Distributed Systems Security (NDSS'99) Symposium, The Internet Society*, 1999.

[2] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. In *Proceedings of WWW9 Conference*, 1999.

[3] H. Chen, F. Perich, D. Chakraborty, T. Finin, and A. Joshi. Intelligent agents meet semantic web in a smart meeting room. In *Proceedings of the third International Joint Conference on Autonomous Agents and Multi Agent Systems*, 2004.

[4] J. Delgado. Emergence of social conventions in comple networks. *Artificial intelligence*, 141:171–185, 2002.

[5] L. Ding, L. Zhou, and T. Finin. Trust based knowledge outsourcing for semantic web agents. In *Proceedings of IEEE/WIC International Conference on Web Intelligence*, 2003.

[6] J. Golbeck, B. Parsia, and J. Hendler. Trust networks on the semantic web. In *Proceedings of Cooperative Intelligent Agents*, 2003.

[7] T. Grandison and M. Sloman. A survey of trust in internet application. *IEEE Communications Surveys & Tutorials (Fourth Quarter)*, 3(4), 2000.

[8] H. A. Kautz, B. Selman, and M. A. Shah. The hidden web. *AI Magazine*, 18(2):27–36, 1997.

[9] J. E. Kittock. The impact of locality and authority on emergent conventions: initial observations. In *Proceedings of the twelfth national conference on Artificial intelligence (vol. 1)*, pages 420–425. American Association for Artificial Intelligence, 1994.

[10] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[11] J. Parker, J. L. Undercoffer, J. Pinkston, and A. Joshi. On intrusion detection in mobile ad hoc networks. In *Workshop on Information Assurance (WIA04)*, 2003. In conjunction with the 23rd IEEE International Performance Computing and Communications Conference (IPCCC).

[12] F. Perich, A. Joshi, T. Finin, , and Y. Yesha. On data management in pervasive computing environments. *IEEE Transaction on knowledge and data engineering*, 16(5):621–634, 2004.

[13] R.Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the Thirteenth International World Wide Web Conference*, 2004.

[14] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the Second International Semantic Web Conference*, 2003.

[15] M. P. Singh, B. Yu, and M. Venkatraman. Community-based service location. *Commun. ACM*, 44(4):49–54, 2001.

[16] R. M. Sreenath and M. P. Singh. Agent-based service selection. *Web Semantics(in press)*, 2003.

[17] P. Yolum and M. P. Singh. Emergent properties of referral systems. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2003.

[18] B. Yu and M. P. Singh. Distributed reputation management for electronic commerce. *Computational Intelligence*, 18(4):535–549, 2002.

[19] B. Yu and M. P. Singh. Searching social networks. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2003.

[20] B. Yu, M. Venkatraman, and M. P. Singh. An adaptive social network for information access: Theoretical and experimental results. *Journal of the Applied Artificial Intelligence*, 17(1), 2003.

[21] Y. Zou, T. Finin, L. Ding, H. Chen, and R. Pan. Using semantic web technology in multi-agent systems: a case study in the taga trading agent environment. In *Proceedings of the 5th international conference on Electronic commerce*, 2003.